UPMEM PIM

The first commercially available PIM-DRAM

Monday In Memory, 7th of June 2021

Fabrice Devaux, CTO @UPMEM

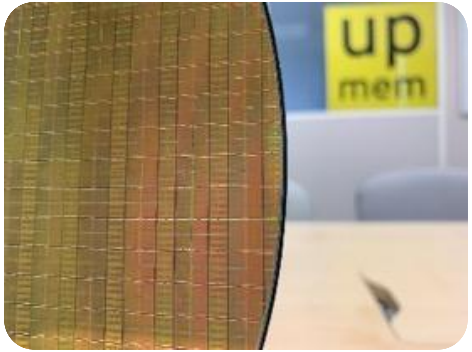
Who am I ?

- UPMEM CTO and Co-founder
- Chief architect of UPMEM's architecture
- Work on PIM since 2013

Contact : fdevaux@upmem.com

The company

- Deep semiconductor & server system expertise - based in Grenoble, France
- Technology & product completed - 1st product - major IPs
- Fabless semiconductor business model deployed
- Deliveries of PIM modules in US, China and Europe
- Active PIMaaS datacenter for customers & labs
- Tens of successful use cases & prime labs collaborations



Some terminology

PIM: different meanings for different people...

In array computing	→	Flash (MYTHIC and others), research on exotic NVM)
In die computing (1)	→	DRAM: UPMEM, SAMSUNG, Neuroblade
Stacked die computing (1)	→	Academic papers around HMC (but HMC now dead)
DIMM level computing	→	Academic papers + some companies toying with the concept
in storage computing	→	Flash (NGD system and others) (regular FLASH chips + FPGA/ASIC)

(1) sometime referred to as 'in memory computing', sometime as 'near memory computing'

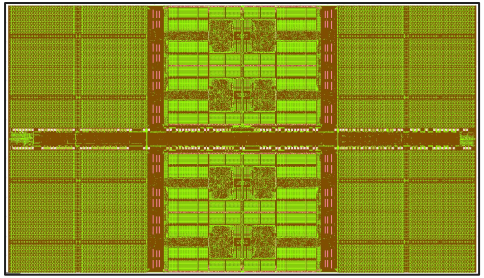
UPMEM PIM DRAM (1/2)

8 x 32-bit CPU added to a 4Gb DRAM die:

- First Gen: 8 x CPU @450MHz, 8 x 64 MB banks (1 CPU for 1 bank)
- Second Gen: 8 x CPU @600MHz, 16 x 32 MB banks (1 CPU for 2 banks), secure Enclave

Multi-threaded CPU:

- In order execution at the thread level
- out of order execution between threads when executing DMA instructions



Offering/Roadmap:

- 1st Gen: 24 hardware threads, scalar → in production
- 2nd Gen: 16 hardware threads, scalar → in design
- 3rd Gen: 32 hardware threads, 2 way superscalar → planning

UPMEM PIM DRAM (2/2)

No instruction cache, but a 24 KB instruction memory

➔ Instructions can be loaded from DRAM through DMA instructions

No data cache, but a 64 KB Work RAM

➔ Data can be loaded/stored from/to DRAM through DMA instructions
➔ DMA instructions do not stop pipeline execution for the non-concerned threads
➔ Caches and threading do not fit well
➔ Caches would be difficult (big & slow) to implement on a DRAM process

A proprietary ISA ?

Retargeting an open source compiler is possible (thanks LLVM)

90% compatibility is not interesting - last 10% could be difficult to achieve due to DRAM process limitation

⟹ Mainstream ISA are designed to cover a very large design space:

- From MCU (scalar in order) to server Multi Core (superscalar out of order)
- Possibilities on DRAM process far more limited

A somewhat **specialized ISA** adapted to constraints, allow for more performances

UPMEM Approach

- ➔ on die processing
- ➔ standard cell synthesis and custom SRAM and register files
- ➔ processing done at a bank granularity
- ➔ unaltered bank
- ➔ only bank connection extended / modified
- ➔ usage of a mainstream protocol (DDR4)

SAMSUNG : same approach for their FIM, Function In memory (HBM2 instead of DDR4)

CAUTION ⚠ **Do you want to do DRAM PIM for real ? Don't mess with the array !**

Key properties

- Scalable ad nauseam
  - Only the host can access the totality of the memory
  - Each DPU can only access its associated memory
- No burden of establishing new interchip communications

- LLVM / CLANG based SDK
- Proprietary ISA for ease of implementation
- C and RUST programmable

- No OS support inside the DPU (no OS needed in the first place)
- BUT Software debug support

Current applications

- Genomics
- Analytics
- Search
- Database

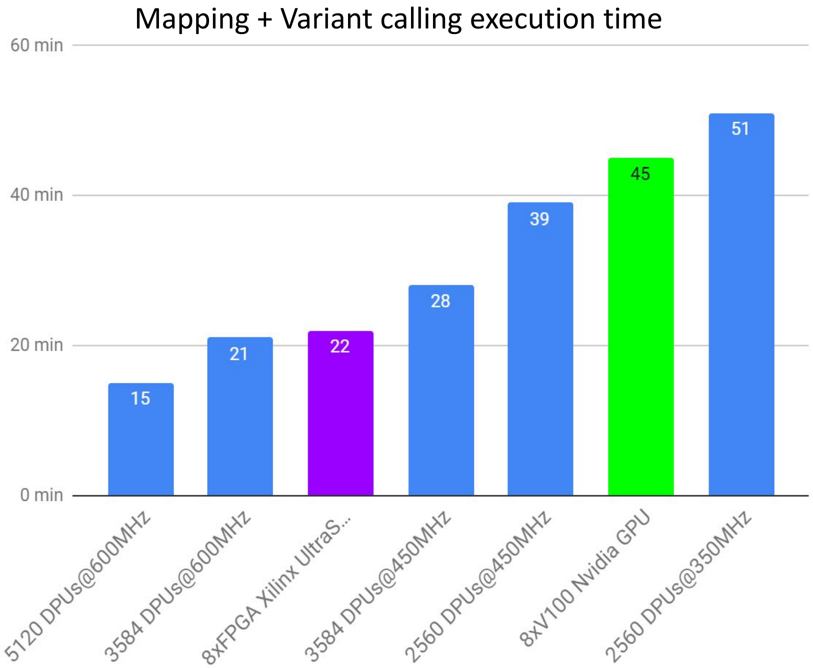
New targeted Applications

- IA
- Security

Genomic Pipeline Acceleration

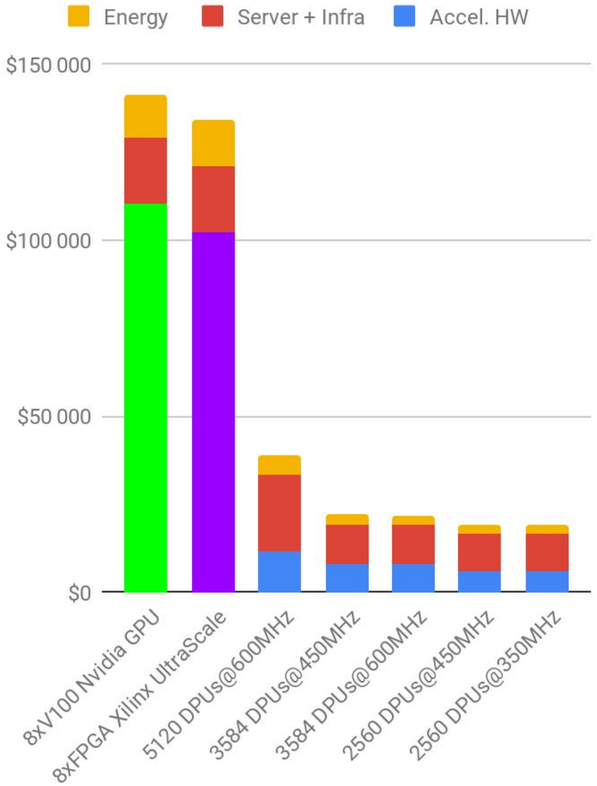
GATK (reference) = ~20h on CPU platforms

- Today UPMEM PIM on par with 8 GPUs (Nvidia solution) at 45
- Tomorrow faster than 8 FPGA (Illumina solution)



Genomic Pipeline TCO consideration

- ➔ 5-7x lower TCO (3 years)
- ➔ 9x less energy consumption than GPUs
- ➔ 5x less energy consuming than FPGAs



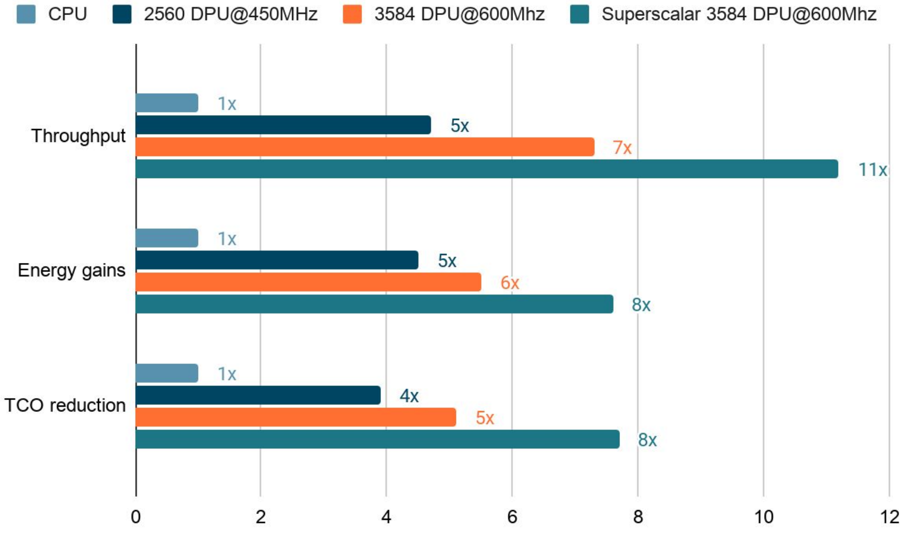
Further considerations

When running the genomic PIM application on 2x x86 4115 + 2560 DPUs @450MHz

- **90%** (~55 MB) **of memory used** by each bank (max 64 MB)
- **0,9 instruction/cycle** for DPUs in average
- **DPUs occupied at 72%** in average
- **160 GB/s of PIM bandwidth** (7% of its full capabilities)
- The **host is used at 21%**

Benefit example: Index Search

- Document database search over a **PIM architecture**
- Partnership with a global US-based search engine leader to improve one of their **worst-case**: a chain of 5 words query in a document database (12M+ text documents, 120GB index)
- Preliminary **Comparisons with Apache Lucene show a 30x acceleration**



Many workloads accelerated

- Genomics (INRIA, SKKU, Industry leaders)
- Image processing + Fourier transforms (NCKU)
- Graph Algorithms, Security (ETH)
- Swapping, search compression/decompression (UBC, UCR)
- In-Memory database (SAP)
- Key Value Store, recommendation algos (UBC, GAFAM)
- Machine Learning (ETH, IBM, UBC, RIT, INPG, US Air Force, Samsung)
- Spacecraft computing (ESA)
- Security (Orange, Morgan State U.)
- Hardware, hybrid architecture (NCSU, EPFL)
- PIM friendly OS & Compilers (U. of Edinburg, Yonsei U.)

Design Technical challenges

- Slow transistors
- Low routing density
- No IP offering
- No standard cells
- No SRAM generators
- No established design flow

System Technical challenges

- BIOS adaptation
- Cache coherency (lake of)
- Undocumented behaviours of big server chips
  - Sometime doing unexpected things
- Power management

What NOT to do on DRAM ? (1/2)

Analog processing (a.k.a. electric charge sharing) on DRAM not feasible

Reason: the charge to share is already barely readable when not shared

- If the charge was easily readable, DRAM designers would reduce the capacity, increasing the density, until the charge becomes barely readable
- The window for analog computing on DRAM closed a long time ago.
- Too much of a technology development and risks to be realistically envisioned.

What NOT to do on DRAM ? (2/2)

Adding logic at the sense amplifier level:

- **DRAM array are always heavily repaired** (no pristine DRAM for more than a decade)
  - Processing logic to deal with repair of a DRAM array is unrealistic
- The **design of sense amplifiers is extremely constrained** due to the pitch of the bit lines they are sampling
- Perimeter overheads
  - Operators spread across the array (first level sense amplifiers) would induce connection overheads
  - A complete CPU is connected only once with its corresponding DRAM array.

Reflexions on technological trends (1/2)

- For 20 years, new non-volatile memory are about to disrupt the market every 6 months
  - In array processing on such technology would be multiplying risk by risk
- A memory capable of accelerating calculation or an accelerator embedding memory ?
  - High density memory enhanced with PIM capability
    - Still useable as a memory in itself
    - UPMEM, SAMSUNG
  - Lower density memory, specialized for a given calculation (IA)
- Just another way of implementing a given acceleration, no longer an actual memory

Reflexions on technological trends (2/2)

- Thermal challenge on processing with stacked logic
  - reliability: thermal cycling fracturing TSV
- price & manufacturability challenge
  - HBM successful, but limited to high end (expensive) market

How to start working with PIM

- ➔ SDK with simulator and documentation are available at:

  https://sdk.upmem.com/

- ➔ Browse code and all PIM use cases on our github:

  https://github.com/upmem/

- ➔ **Start your project on our PIM datacenter or order your own PIM DIMMs**

  contact@upmem.com